Microsoft SQL Server Dynamic Data Masking Policies

Solutions for **Success**

POINT ALLIANCE

Dynamic Data Masking

- Introduced in SQL Server 2016
- Used to hide data from the user on the client-side
- Not the same as data encryption

Why use Dynamic Data Masking

- A call center support person may identify callers by several digits of their social security number or credit card number. Social security numbers or credit card numbers should not be fully exposed to the support person.
  - A masking rule can be defined that masks all but the last four digits of any social security number or credit card number in the result set of any query.
- For another example, by using the appropriate data mask to protect personally identifiable information (PII) data, a developer can query production environments for troubleshooting purposes without violating compliance regulations.

Dynamic Data Masking

- Greatly simplifies the design and coding of security in your application
- Helps prevent unauthorized access to sensitive data by enabling customers to specify how much sensitive data to reveal with minimal impact on the application layer
- Limits sensitive data exposure by masking it to non-privileged users
- can be configured on designated database fields to hide sensitive data in the result sets of queries

Masking Function: Default

- Full masking according to the data types of the designated fields
  - Use XXXX or fewer Xs if the size of the field is less than 4 characters for string data types (nchar, ntext, nvarchar).
  - Use a zero value for numeric data types (bigint, bit, decimal, int, money, numeric, smallint, smallmoney, tinyint, float, real).
  - Use 01-01-1900 for date/time data types (date, datetime2, datetime, datetimeoffset, smalldatetime, time).
  - For SQL variant, the default value of the current type is used.
  - For XML the document <masked/> is used.
  - Use an empty value for special data types (timestamp table, hierarchyid, GUID, binary, image, varbinary spatial types).

Masking Function: Email

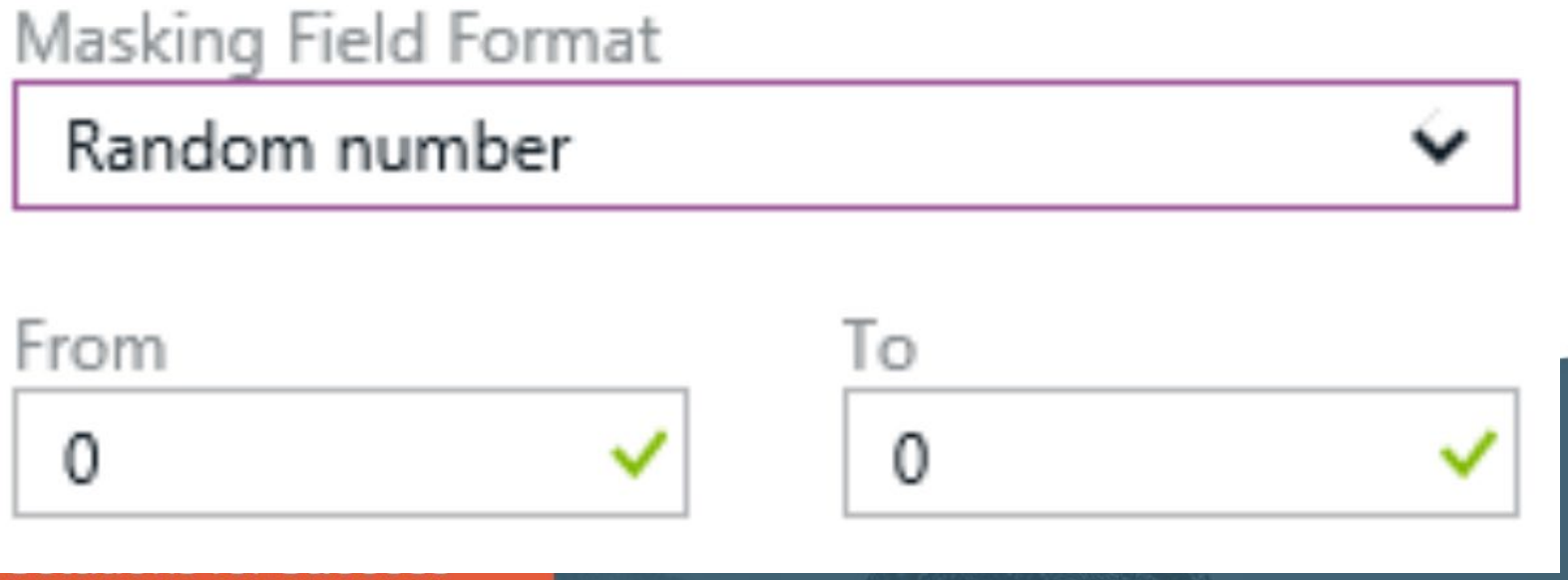
- Masking method, which exposes the first letter and replaces the domain with XXX.com using a constant string prefix in the form of an email address.
- aXX@XXXX.com

Masking Function: Credit Card

- Masking method, which exposes the last four digits of the designated fields and adds a constant string as a prefix in the form of a credit card.
- XXXX-XXXX-XXXX-1234

Masking Function: Random Number

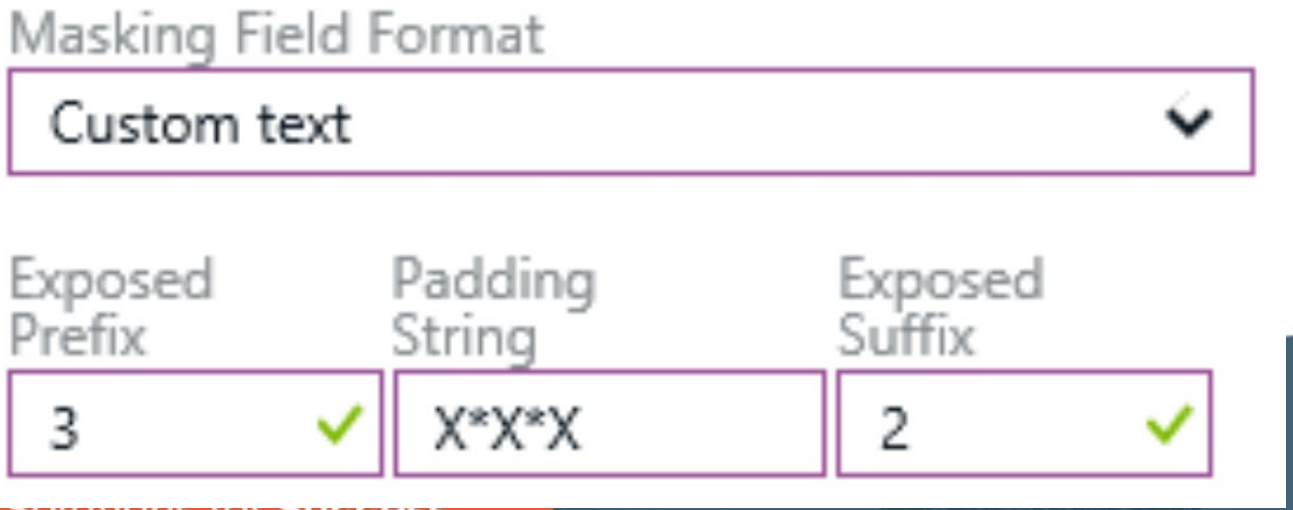
- Masking method, which generates a random number according to the selected boundaries and actual data types. If the designated boundaries are equal, then the masking function is a constant number.



POINT ALLIANCE

Masking Function: Custom Text

- **Masking method, which exposes the first and last characters** and adds a custom padding string in the middle. If the original string is shorter than the exposed prefix and suffix, only the padding string is used.
  prefix[padding]suffix



Example

- CREATE TABLE Employee_Financial (
- Emp_ID INT IDENTITY(1, 1) PRIMARY KEY
- ,Emp_First_Name NVARCHAR(10) NOT NULL
- ,Emp_Last_Name NVARCHAR(10) NOT NULL
- ,Emp_Date_Of_Birth DATETIME NULL
- ,Emp_Salary INT NULL
- ,Emp_Email NVARCHAR(50) NULL
- ,Emp_Employment_Date DATETIME NULL
-)

- CREATE USER DDMUser WITHOUT LOGIN;
- GRANT SELECT ON Employee_Financial TO DDMUser;

Emp_ID	Emp_First_Name	Emp_Last_Name	Emp_Date_Of_Birth	Emp_Salary	Emp_Email	Emp_Employment_Date
1	Jerome	Martinez	1981-07-17 01:34:14.000	2598	ginkm.whcjdh@uvv.org	1995-08-28 01:22:48.000
2	Roland	Garcia	1989-10-13 02:02:51.000	2036	qztdq.jibjm@grr.co	1994-08-13 04:23:04.000
3	Ernest	Smith	1980-10-18 17:29:27.000	1332	rfqoeu.nslyt@qtl.com	2002-07-27 03:34:44.000
4	Jorge	Davis	1984-09-09 04:31:38.000	1666	scwzvxp.twyujf@keb.org	2002-11-16 15:17:01.000
5	Marvin	Williams	1990-08-31 13:00:52.000	872	eojtrq.ukzetgc@poxc.org	1998-01-14 12:19:23.000
6	Stella	Wilson	1979-10-26 18:11:24.000	2046	stcub.syidn@droe.com	1995-08-17 12:09:45.000
7	Salvador	Taylor	1982-11-29 06:07:52.000	1278	ipcmwy.mfxes@gzdw.com	1995-08-11 14:02:47.000
8	Aaliyah	Miller	1993-06-17 15:00:30.000	705	iybupw.ofbfo@ndx.org	2015-11-19 19:12:36.000
9	Lawrence	Thomas	1984-10-11 15:51:25.000	1745	qnruq.uiimjb@hjhhj.co	1991-07-07 15:07:18.000
10	Nicholas	Rodriguez	1983-04-24 18:11:34.000	2772	cmsqh.nwwyoe@hniybtq.org	1996-07-31 13:25:07.000
11	Alex	Anderson	1986-01-15 04:45:37.000	2090	vgqqc.hjtuu@mbyb.co	2014-05-05 18:24:51.000
12	Ray	Hernandez	1990-12-06 00:40:41.000	1871	pipiwrdy.xvzmfer@fmt.org	2007-07-06 16:58:06.000
13	Gilbert	Jones	1994-06-22 20:36:18.000	2339	ululey.swmqwri@lfu.com	2007-01-24 00:36:38.000
14	Aria	Brown	1996-02-20 15:36:23.000	1095	yhfgit.nqffh@phvy.com	2012-10-20 02:46:23.000
15	Edward	Johnson	1991-12-26 04:01:08.000	1569	kapxts.cmpxw@frs.org	2006-08-14 10:39:27.000

Default Data Masking Example

- ALTER TABLE Employee_Financial
- ALTER COLUMN EMP_Last_Name varchar(10) MASKED WITH (FUNCTION = 'default()');

Default Data Masking Example

	Emp_ID	Emp_First_Name	Emp_Last_Name	Emp_Date_Of_Birth	Emp_Salary	Emp_Email	Emp_Employment_Date
1	1	Jerome	xxxx	1981-07-17 01:34:14.000	2598	ginkm.whcjdh@uvv.org	1995-08-28 01:22:48.000
2	2	Roland	xxxx	1989-10-13 02:02:51.000	2036	qztdq.jibjm@grr.co	1994-08-13 04:23:04.000
3	3	Ernest	xxxx	1980-10-18 17:29:27.000	1332	rfqoeu.nslyt@qtl.com	2002-07-27 03:34:44.000
4	4	Jorge	xxxx	1984-09-09 04:31:38.000	1666	scwzvxp.twyujf@keb.org	2002-11-16 15:17:01.000
5	5	Marvin	xxxx	1990-08-31 13:00:52.000	872	eojtrq.ukzetgc@poxc.org	1998-01-14 12:19:23.000
6	6	Stella	xxxx	1979-10-26 18:11:24.000	2046	stcub.syidn@droe.com	1995-08-17 12:09:45.000
7	7	Salvador	xxxx	1982-11-29 06:07:52.000	1278	ipcmwy.mfxes@gzdw.com	1995-08-11 14:02:47.000
8	8	Aaliyah	xxxx	1993-06-17 15:00:30.000	705	iybupw.ofbfo@ndx.org	2015-11-19 19:12:36.000
9	9	Lawrence	xxxx	1984-10-11 15:51:25.000	1745	qnruq.uiimjb@hjhhj.co	1991-07-07 15:07:18.000
10	10	Nicholas	xxxx	1983-04-24 18:11:34.000	2772	cmsqh.nwwyoe@hniybtq.org	1996-07-31 13:25:07.000
11	11	Alex	xxxx	1986-01-15 04:45:37.000	2090	vgqqc.hjtuu@mbyb.co	2014-05-05 18:24:51.000
12	12	Ray	xxxx	1990-12-06 00:40:41.000	1871	pipiwrdy.xvzmfer@fmt.org	2007-07-06 16:58:06.000
13	13	Gilbert	xxxx	1994-06-22 20:36:18.000	2339	ululey.swmqwri@tfu.com	2007-01-24 00:36:38.000
14	14	Aria	xxxx	1996-02-20 15:36:23.000	1095	yhfgit.nqffh@phvy.com	2012-10-20 02:46:23.000
15	15	Edward	xxxx	1991-12-26 04:01:08.000	1569	kapxts.cmpxw@frs.org	2006-08-14 10:39:27.000

Email Data Masking Example

- ALTER TABLE Employee_Financial
- ALTER COLUMN EMP_Email nvarchar(50) MASKED WITH (FUNCTION = 'Email()');

Email Data Masking Example

	Emp_ID	Emp_First_Name	Emp_Last_Name	Emp_Date_Of_Birth	Emp_Salary	Emp_Email	Emp_Employment_Date
1	1	Jerome	xxxx	1981-07-17 01:34:14.000	2598	gXXX@XXXX.com	1995-08-28 01:22:48.000
2	2	Roland	xxxx	1989-10-13 02:02:51.000	2036	qXXX@XXXX.com	1994-08-13 04:23:04.000
3	3	Ernest	xxxx	1980-10-18 17:29:27.000	1332	rXXX@XXXX.com	2002-07-27 03:34:44.000
4	4	Jorge	xxxx	1984-09-09 04:31:38.000	1666	sXXX@XXXX.com	2002-11-16 15:17:01.000
5	5	Marvin	xxxx	1990-08-31 13:00:52.000	872	eXXX@XXXX.com	1998-01-14 12:19:23.000
6	6	Stella	xxxx	1979-10-26 18:11:24.000	2046	sXXX@XXXX.com	1995-08-17 12:09:45.000
7	7	Salvador	xxxx	1982-11-29 06:07:52.000	1278	iXXX@XXXX.com	1995-08-11 14:02:47.000
8	8	Aaliyah	xxxx	1993-06-17 15:00:30.000	705	iXXX@XXXX.com	2015-11-19 19:12:36.000
9	9	Lawrence	xxxx	1984-10-11 15:51:25.000	1745	qXXX@XXXX.com	1991-07-07 15:07:18.000
10	10	Nicholas	xxxx	1983-04-24 18:11:34.000	2772	cXXX@XXXX.com	1996-07-31 13:25:07.000
11	11	Alex	xxxx	1986-01-15 04:45:37.000	2090	vXXX@XXXX.com	2014-05-05 18:24:51.000
12	12	Ray	xxxx	1990-12-06 00:40:41.000	1871	pXXX@XXXX.com	2007-07-06 16:58:06.000
13	13	Gilbert	xxxx	1994-06-22 20:36:18.000	2339	uXXX@XXXX.com	2007-01-24 00:36:38.000
14	14	Aria	xxxx	1996-02-20 15:36:23.000	1095	yXXX@XXXX.com	2012-10-20 02:46:23.000
15	15	Edward	xxxx	1991-12-26 04:01:08.000	1569	kXXX@XXXX.com	2006-08-14 10:39:27.000

Random Data Masking Example

- ALTER TABLE Employee_Financial
- ALTER COLUMN EMP_Salary int MASKED WITH (FUNCTION='random(1,9)');

Random Data Masking Example

	Emp_ID	Emp_First_Name	Emp_Last_Name	Emp_Date_Of_Birth	Emp_Salary	Emp_Email	Emp_Employment_Date
1	1	Jerome	xxxx	1981-07-17 01:34:14.000	9	gXXX@XXXX.com	1995-08-28 01:22:48.000
2	2	Roland	xxxx	1989-10-13 02:02:51.000	2	qXXX@XXXX.com	1994-08-13 04:23:04.000
3	3	Ernest	xxxx	1980-10-18 17:29:27.000	9	rXXX@XXXX.com	2002-07-27 03:34:44.000
4	4	Jorge	xxxx	1984-09-09 04:31:38.000	5	sXXX@XXXX.com	2002-11-16 15:17:01.000
5	5	Marvin	xxxx	1990-08-31 13:00:52.000	4	eXXX@XXXX.com	1998-01-14 12:19:23.000
6	6	Stella	xxxx	1979-10-26 18:11:24.000	7	sXXX@XXXX.com	1995-08-17 12:09:45.000
7	7	Salvador	xxxx	1982-11-29 06:07:52.000	1	iXXX@XXXX.com	1995-08-11 14:02:47.000
8	8	Aaliyah	xxxx	1993-06-17 15:00:30.000	4	iXXX@XXXX.com	2015-11-19 19:12:36.000
9	9	Lawrence	xxxx	1984-10-11 15:51:25.000	9	qXXX@XXXX.com	1991-07-07 15:07:18.000
10	10	Nicholas	xxxx	1983-04-24 18:11:34.000	1	cXXX@XXXX.com	1996-07-31 13:25:07.000
11	11	Alex	xxxx	1986-01-15 04:45:37.000	1	vXXX@XXXX.com	2014-05-05 18:24:51.000
12	12	Ray	xxxx	1990-12-06 00:40:41.000	5	pXXX@XXXX.com	2007-07-06 16:58:06.000
13	13	Gilbert	xxxx	1994-06-22 20:36:18.000	5	uXXX@XXXX.com	2007-01-24 00:36:38.000
14	14	Aria	xxxx	1996-02-20 15:36:23.000	2	yXXX@XXXX.com	2012-10-20 02:46:23.000
15	15	Edward	xxxx	1991-12-26 04:01:08.000	7	kXXX@XXXX.com	2006-08-14 10:39:27.000

Custom Data Masking Example

- ALTER TABLE Employee_Financial
- ALTER COLUMN EMP_First_name nvarchar(10) MASKED WITH (FUNCTION= 'partial(3,"XXXX",3)');
-

Custom Data Masking Example

	Emp_ID	Emp_First_Name	Emp_Last_Name	Emp_Date_Of_Birth	Emp_Salary	Emp_Email	Emp_Employment_Date
1	1	XXXX	xxxx	1981-07-17 01:34:14.000	9	gXXX@XXXX.com	1995-08-28 01:22:48.000
2	2	XXXX	xxxx	1989-10-13 02:02:51.000	1	qXXX@XXXX.com	1994-08-13 04:23:04.000
3	3	XXXX	xxxx	1980-10-18 17:29:27.000	1	rXXX@XXXX.com	2002-07-27 03:34:44.000
4	4	XXXX	xxxx	1984-09-09 04:31:38.000	5	sXXX@XXXX.com	2002-11-16 15:17:01.000
5	5	XXXX	xxxx	1990-08-31 13:00:52.000	5	eXXX@XXXX.com	1998-01-14 12:19:23.000
6	6	XXXX	xxxx	1979-10-26 18:11:24.000	2	sXXX@XXXX.com	1995-08-17 12:09:45.000
7	7	SalXXXXdor	xxxx	1982-11-29 06:07:52.000	7	iXXX@XXXX.com	1995-08-11 14:02:47.000
8	8	AalXXXXyah	xxxx	1993-06-17 15:00:30.000	4	iXXX@XXXX.com	2015-11-19 19:12:36.000
9	9	LawXXXXnce	xxxx	1984-10-11 15:51:25.000	9	qXXX@XXXX.com	1991-07-07 15:07:18.000
10	10	NicXXXXlas	xxxx	1983-04-24 18:11:34.000	7	cXXX@XXXX.com	1996-07-31 13:25:07.000
11	11	XXXX	xxxx	1986-01-15 04:45:37.000	9	vXXX@XXXX.com	2014-05-05 18:24:51.000
12	12	XXXX	xxxx	1990-12-06 00:40:41.000	1	pXXX@XXXX.com	2007-07-06 16:58:06.000
13	13	GilXXXXert	xxxx	1994-06-22 20:36:18.000	6	uXXX@XXXX.com	2007-01-24 00:36:38.000
14	14	XXXX	xxxx	1996-02-20 15:36:23.000	3	yXXX@XXXX.com	2012-10-20 02:46:23.000
15	15	XXXX	xxxx	1991-12-26 04:01:08.000	5	kXXX@XXXX.com	2006-08-14 10:39:27.000

Query Masked Column Definitions

- SELECT TBLS.name as TableName,MC.NAME ColumnName, MC.is_masked IsMasked, MC.masking_function MaskFunction
- FROM sys.masked_columns AS MC
- JOIN sys.tables AS TBLS
- ON MC.object_id = TBLS.object_id
- WHERE is_masked = 1;

	TableName	ColumnName	IsMasked	MaskFunction
1	Employee_Financial	Emp_First_Name	1	partial(3, "XXXX", 3)
2	Employee_Financial	Emp_Last_Name	1	default()
3	Employee_Financial	Emp_Salary	1	random(1, 9)
4	Employee_Financial	Emp_Email	1	email()

Remove Data Masking

- ALTER TABLE Employee_Financial
- ALTER COLUMN EMP_First_name DROP MASKED;
-

	Emp_ID	Emp_First_Name	Emp_Last_Name	Emp_Date_Of_Birth	Emp_Salary	Emp_Email	Emp_Employment_Date
1	1	Jerome	xxxx	1981-07-17 01:34:14.000	9	gXXX@XXXX.com	1995-08-28 01:22:48.000
2	2	Roland	xxxx	1989-10-13 02:02:51.000	2	qXXX@XXXX.com	1994-08-13 04:23:04.000
3	3	Ernest	xxxx	1980-10-18 17:29:27.000	9	rXXX@XXXX.com	2002-07-27 03:34:44.000
4	4	Jorge	xxxx	1984-09-09 04:31:38.000	5	sXXX@XXXX.com	2002-11-16 15:17:01.000
5	5	Marvin	xxxx	1990-08-31 13:00:52.000	4	eXXX@XXXX.com	1998-01-14 12:19:23.000
6	6	Stella	xxxx	1979-10-26 18:11:24.000	7	sXXX@XXXX.com	1995-08-17 12:09:45.000
7	7	Salvador	xxxx	1982-11-29 06:07:52.000	1	iXXX@XXXX.com	1995-08-11 14:02:47.000
8	8	Aaliyah	xxxx	1993-06-17 15:00:30.000	4	iXXX@XXXX.com	2015-11-19 19:12:36.000
9	9	Lawrence	xxxx	1984-10-11 15:51:25.000	9	qXXX@XXXX.com	1991-07-07 15:07:18.000
10	10	Nicholas	xxxx	1983-04-24 18:11:34.000	1	cXXX@XXXX.com	1996-07-31 13:25:07.000
11	11	Alex	xxxx	1986-01-15 04:45:37.000	1	vXXX@XXXX.com	2014-05-05 18:24:51.000
12	12	Ray	xxxx	1990-12-06 00:40:41.000	5	pXXX@XXXX.com	2007-07-06 16:58:06.000
13	13	Gilbert	xxxx	1994-06-22 20:36:18.000	5	uXXX@XXXX.com	2007-01-24 00:36:38.000
14	14	Aria	xxxx	1996-02-20 15:36:23.000	2	yXXX@XXXX.com	2012-10-20 02:46:23.000
15	15	Edward	xxxx	1991-12-26 04:01:08.000	7	kXXX@XXXX.com	2006-08-14 10:39:27.000

Copying and Exporting Data

- Using SELECT INTO , INSERT INTO or SQL Server Import and Export wizard methods to copy the masked data to another table, the user will be presented with masked at the destination table.

Downside

- Dynamic data masking has its shortcomings. Stored procedures can’t be dynamically masked because their execution algorithms are stored within the database and client applications just request the execution according to an already existing plan. Thus, masking of stored procedures requires rewriting the query results, not the query itself.

Any Questions?